IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 23, 2020, accepted October 31, 2020, date of publication November 3, 2020, date of current version November 19, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.3035720*

# A Deep-Learning-Based Method for Solving Nonlinear Singular Lane-Emden Type Equation

**JING HE[1], PO LONG[1], XINYING WANG[2], (Member, IEEE), AND KUN HE[3]**
[1]Software College, Changsha Social Work College, Changsha 410004 China
[2]Global Energy Interconnection Research Institute, Beijing 102209, China
[3]China Electric Power Research Institute, Beijing 100192, China

Corresponding author: Kun He (kh880622@126.com)

This work was supported in part by the Natural Science Funding of China under Grant 51607162.

**ABSTRACT** The nonlinear Lane-Emden type equation can be used to describe many physical phenomena. To solve this type of equation, a method based on deep neural network is proposed. The output layer of this network has two layers, the last one of which scaling the outputs of their neighbors with an aim at coping with issues where the values of function to be approximated are much less or larger than the order of 1. The Lane-Emden equation and its initial conditions are employed to construct the loss function, and the problem of solving Lane-Emden equation is transformed into an optimization problem. A hybrid method combined Adam and L-BFGS-B methods is used to solve the optimization problem and consequently the Lane-Emden type equation is solved. To increase the accuracy, an adaptive strategy is incorporated into the training data sampling method. Especially, a strategy in coping with the issue about solving white-dwarf problem is proposed. Numerical experiments are conducted in which reference solutions including analytical solutions and numerical solutions given by Runge-Kutta method are used to verify the effectiveness of our proposed method. More importantly, the solutions over large domains are calculated by the use of the proposed method. The results show that the results given by our method are in good agreement with the reference solutions, and in cases where existent neural-network-based method fails, our method is able to produce convincible results.

**INDEX TERMS** Deep learning, artificial neural network, Lane-Emden type equation, polytropic gas sphere model, isothermal gas sphere model, white-dwarf model.

## I. INTRODUCTION

The Lane-Emden equation was named after two physicists Jonathan H. Lane and Robert Emden in memory of their first use of this type of equation in investigating the equilibrium density distribution in self-gravitating spherical polytropic gas sphere in dytrostatic equilibrium [1], [2]. Later it was modified to model isothermal gas spheres in gravitational equilibrium [3], the gravitational potential of degenerate white-dwarf star [3], the formation and propagation of shock formed during gravitational collapse of polytropic gas sphere [4]. Besides its applications in astrophysics, this type of equation was employed to simulate thermal explosion in cylindrical vessel [5] thermal distribution in human head [6] and stress on an axisymmetric shallow membrane cap [7].

The main difficulty to solve Lane-Emden equation numerically is the singularity at the point $x = 0$. To solve this type of equation, some methods including analytical ones, numerical ones and their combinations have been proposed in the literature. For analytical methods, Adomian decomposition method [8], [9], homotopy method [10]–[15], and transformation method [16] are developed. Adomian decomposition method was first proposed to solve polytropic gas sphere problem in [8] where the Lane-Emden equation is reformulated into a form of operator equation onto which Adomian decomposition is applied and a recursive relation of the components in the series expansion of the solution can be derived. This method was further improved in [9] where a well-defined transformation was used to remove the term $y'$ in the equation before the use of Adomian decomposition. Sometimes it is troublesome to calculate the Adomian polynomials in the Adomian decomposition method. A universal framework of homotopy analysis method for differential equation was first proposed in [10] and possibly first used to solve Lane-Emden equation in [11]. This method can be considered as a generalized Taylor expansion method.

The associate editor coordinating the review of this manuscript and approving it for publication was Jin-Liang Wang.

An improved homotopy analysis method was applied to solve the Lane-Emden equation in [12]. The advantage of this method is its built-in convergence criteria similar to the original one but more flexible. In addition to homotopy analysis method, homotopy perturbation method was first proposed in [13] and later proposed to tackle the Lane-Emden equation in [14], [15]. It is able to use low-order approximation solution to obtain high accuracy, while the homotopy analysis method needs more terms in the series to achieve the same accuracy. It is also shown in [15] that Adomian's method may be interpreted as a homotopy perturbation method. It seems that the transformation method in [16] gives solution differs from the numerical solutions for the polytropic gas sphere problem with polytropic indices $m = 2$ and 3. One of the shortcomings of the analytical methods is that many terms of complex forms which are hard to deduce are needed to be included into the series expansion to increase the accuracy and it is difficult to find approximated formulae for large domains.

With regard to numerical methods, Runge-Kutta method [4], [17], wavelet method [18]–[20], collocation method [21]–[24], operational matrix of differentiation method [25]–[28], squared remainder minimization method [29], variational iteration method [30] and Finite Difference Method are designed [2]. Runge-Kutta method is one of the classical methods for solving differential equations. The standard Runge-Kutta method fails to start integration because of the singularity at the origin. One way to circumvent this problem is to integrate the problem from some point a little more right from origin, which is very efficient in practice. An improvement was made in [17] to cope with the singularity at the origin. In [18]–[20], the Lane-Emden equation was transformed into an integral form, and Legendre wavelets, ultraspherical wavelets and Haar wavelets were used to approximate the solution, respectively. In [21]–[24], transformed Hermite functions, Lagrange interpolation polynomials, modified generalized Laguerre polynomials and Legendre scaling functions were employed to approximate the solution and linear algebraic systems at a number of collocation points were formed to obtain the expansion coefficients, respectively. In [25]–[28], Bernstein polynomials, Legendre polynomials, second kind Chebyshev polynomials and shifted ultraspherical were used to approximate the solution, respectively, and linear algebraic systems were formed by Petrov-Galerkin method. In [29], polynomials were used to approximate the solution and the squared reminder minimization method was proposed to obtain the coefficients. In [30], the error function of the equation was used to construct a variational iteration scheme and it was proven that the iterative solution tends to the true solution as the number of iteration goes to infinity.

There are some combinations of the forgoing methods. Roul hybrided Adomian decomposition method and collocation method [31]. Singh and Verma combined variational iteration method and homotopy method [32]. Maheshwar and Pratibha merged Green function approach and Adomian decomposition method [33].

In addition to these foregoing deterministic approaches, several stochastic ones using artificial neural networks (ANN) have been brought forward. The advantages of the ANN methods are that it does not need to discretize the differential operator as does the numerical methods; it gives the solution in closed analytic form and the obtained solution is differentiable in the entire domain. Hadian-Rasanan *et al.* proposed to use fractional order of Legendre polynomials as activation function to construct orthogonal Neural Network [34]. Verma and Kumar used sigmoid function to design network [35]. Sabir *et al.* employed Morlet function as activation function to engineer wavelet neural network [36]. All these works use a linear combination of basis functions and have demonstrated promising competence of artificial networks in solving Lane-Emden type equation in a domain of length $L = 1$. However, there are some drawbacks. Firstly, their adoption of single hidden layer could lead to an insufficient approximation ability when the domain is large. Increasing the width might be a choice, but essentially it is still using a linear combination of basis functions to approximate the solution to Lane-Emden type equation. Instead, we propose to increase the depth of network in this article. This is in fact utilizing a composition of basis functions to approximate the solution. Besides, Lane-Emden equations of large domains are frequently encountered in practical applications, and in these cases, function values might be larger than 100 at some subdomains. This cannot be handled by the forgoing networks since it is a common practice that it is hard for networks of three layers (namely one input layer, one hidden layer and one output layer) to approximate functions having values much larger than the order of 1. Secondly, in [34], [35] trial solutions that includes a neural network need to be carefully designed according to the initial conditions. It means one has to change the form of trial solution if the initial condition is modified, which is inconvenient in practical uses. Thirdly, the training data points are equidistance [35], [36] or specially chosen according to the activation function [34]. These are strong constraints on selecting training points, for in some cases training data points should be dense in some regions.

In addition to the foregoing issues, there is one more issue with the existent ANN-based methods. It is difficult to use them to solve the white-dwarf problem in which the term $y^2(x)$–$C$ could be less than 0 for these methods are stochastic, making $(y^2(x)–C)^{3/2}$ be complex values. Here $y(x)$ is the solution and $C$ is a given constant. This issue cannot be directly handled by ANN and also complex $y(x)$ is unphysical. This might be the reason why no report on solving the white-dwarf problem by the existent ANN-based methods was seen in the literature yet.

We notice that deep learning has been successfully used in image processing [37]–[39], natural language processing [40], topology optimization [41], traffic control [42], and many other fields with the help of the powerful approximation ability. We also notice that deep learning has been applied to solve partial differential equations including Schrodinger equation [43], [45], Allen–Cahn equation [43],

Navier–Stokes equation [43], Korteweg–de Vries equation [43], Hamilton–Jacobi–Bellman [44], Burgers' equation [44] and Poisson's equation [45].

The objective of this article is to apply deep neural network into solving Lane-Emden type equation and demonstrates its applicability and good performance in cases where the single layer neural networks employed in the previous works [34]–[36] fail. A deep neural network with one a layer functioning as a scale-controller is proposed. Similar to the existent works, the Lane-Emden type equations and initial conditions are used to form a loss function. But what different is an adaptive training data sampling method is employed to increase the accuracy. More importantly, a strategy to cope with the issue with solving white-dwarf problem is put forward to ensure the solution is physical. The proposed deep neural network method is validated in three kinds of Lane-Emden equations, including polytropic problem, isothermal gas sphere problem and white-dwarf problem by numerical experiments.

The contribution of this article is six-fold:

(1) The second-order Lane-Emden type equation is transformed into a system of two differential equations of first order, and a deep neural network is proposed to solve this system. This is different from all foregoing ANN-based methods, since they are directly using ANN to solving the Lane-Emden type equation. Besides, our network is specially designed for this type of system, as the output layer of this network has two layers, the last one of which scaling up or down the outputs of their neighbors.

(2) Activation functions in our network is user-defined and the training data points are randomly chosen in the domain. In addition, deep learning techniques such as tensor computation and automatic differentiation technique are used in the network training. These improvements have not been reported in the previous works [34]–[36] focusing on solving Lane-Emden equation.

(3) To increase the accuracy, an adaptive strategy is incorporated into the training data sampling method.

(4) A strategy to cope with the issue with solving white-dwarf problem is proposed.

(5) The solutions over large domains are calculated by use of our proposed method. These forgoing three improvements have not been reported in aforementioned works.

(6) The superiority of the proposed deep neural network over single layer neural network is demonstrated in some cases.

## II. THE LANE-EMDEN EQUATIONS

The general form of nonlinear Lane-Emden type equations is formulated as

$$\begin{cases} y''(x) + \dfrac{\alpha}{x} y'(x) + f(x, y(x)) = g(x) \\ x \geq 0, \quad y(0) = c_1, \quad y'(0) = c_2 \end{cases} \tag{1}$$

where $\alpha$, $c_1$ and $c_2$ are constants; $f(x, y)$ is a continuous real valued function and $g(x) \in C[0,1]$ [27]. All of them are given by physical problems. For polytropic gas sphere problem, we have [3]

$$\begin{cases} y''(x) + \dfrac{2}{x} y'(x) + y^m(x) = 0 \\ x \geq 0, \quad y(0) = 1, \quad y'(0) = 0 \end{cases} \tag{2}$$

where $m$ is the polytropic index. When $m = 0$, 1 and 5, equation (2) has analytical solutions given as [3]

$$\begin{cases} y(x) = 1 - \dfrac{1}{6}x^2, & m = 0 \\ y(x) = \dfrac{\sin x}{x}, & m = 1 \\ y(x) = \dfrac{1}{\sqrt{1 + x^2/3}}, & m = 5 \end{cases} \tag{3}$$

For isothermal gas sphere problem, we have [3]

$$\begin{cases} y''(x) + \dfrac{2}{x} y'(x) + e^{y(x)} = 0 \\ x \geq 0, \quad y(0) = 0, \ y'(0) = 0 \end{cases} \tag{4}$$

This equation does not have an analytical solution in the whole domain, but an approximate analytical solution in subdomains were found by Wazwaz [9]

$$y(x) \approx -\frac{x^2}{6} + \frac{x^4}{5 \times 4!} - \frac{8x^6}{21 \times 6!} + \frac{122x^8}{81 \times 8!} - \frac{4087x^{10}}{495 \times 10!} \tag{5}$$

which is valid in the interval [0, 2.2]. This interval is determined by using the numerical solutions given by Runge-Kutta method as references.

For white-dwarf star problem, we have [3]

$$\begin{cases} y''(x) + \dfrac{2}{x} y'(x) + \left(y^2(x) - C\right)^{3/2} = 0 \\ x \geq 0, \quad y(0) = 1, \quad y'(0) = 0 \end{cases} \tag{6}$$

where $C$ is constant and $0 \leq C \leq 1$. This equation does not have an analytical solution in the whole domain either, and an approximate analytical solution in subdomains were found by Chandrasekhar [3]

$$\begin{aligned} y(x) \approx 1 &- \frac{1}{6}q^3x^2 + \frac{1}{40}q^4x^3 - \frac{1}{7!}q^5\left(5q^2 + 14\right)x^6 \\ &+ \frac{1}{3 \times 9!}q^6\left(339q^2 + 280\right)x^8 \\ &+ \frac{1}{5 \times 11!}q^7\left(1425q^4 + 11436q^2 + 4256\right)x^{10} \end{aligned} \tag{7}$$

where $q = \sqrt{1 - C}$ and which is valid in the interval [0, 1.2].

## III. DEEP NEURAL NETWORK METHOD FOR LANE-EMDEN TYPE EQUATION

### A. METHOD DESCRIPTION

To solve the Lane-Emden type equation, a calculation method based on deep neural network is proposed. Firstly, the Lane-Emden equation is transformed into a system of differential equations of first order, which is given as

$$\begin{cases} y'(x) - y_1(x) = 0 \\ y_1'(x) + \frac{\alpha}{x} y_1(x) + f(x, y(x)) - g(x) = 0 \\ y(0) = c_1, \quad (0) = c_2 \end{cases} \tag{8}$$

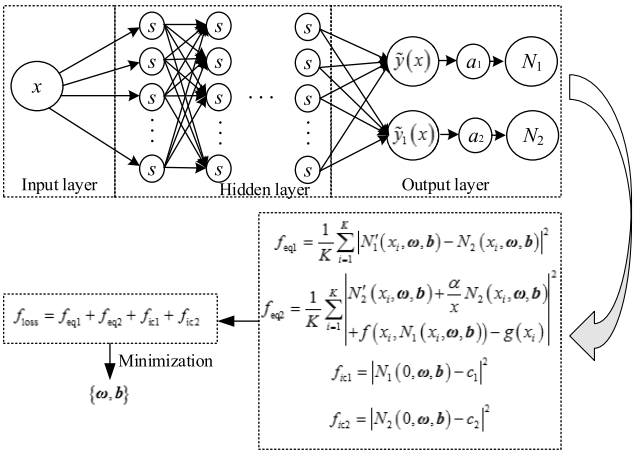



**FIGURE 1.** The structure of the deep neural network.

It is assumed that the solutions $y(x)$ and $y_1(x)$ to equation (8) can be expressed by a deep neural network $\boldsymbol{N}(x, \boldsymbol{\omega}, \boldsymbol{b})$ where $\boldsymbol{\omega}$ is a weight vector formed by all the weights in the network and $\boldsymbol{b}$ is a bias vector consisting of all the biases in the network. The network has one input layer, $L$ hidden layers and two output layers with two output terminals generating approximate function values of $y(x)$ and $y_1(x)$ respectively. These two output terminals are denoted by $N_1(x, \boldsymbol{\omega}, \boldsymbol{b})$ and $N_2(x, \boldsymbol{\omega}, \boldsymbol{b})$ respectively. We have

$$y(x) \approx N_1(x, \boldsymbol{\omega}, \boldsymbol{b}) \tag{9}$$

$$y_1(x) \approx N_2(x, \boldsymbol{\omega}, \boldsymbol{b}) \tag{10}$$

The network used in this article is a fully connected one. The architecture of the network is presented in Figure 1. This network consists of one input layer with one single input terminal, $L$ layers in the hidden layer with activation function $s$ and two layers in the output layer. All activation functions are the same and user-defined. The width $W$ of all layers in the hidden layer are the flexible and user-controlled. In the output layer, the two outputs from the hidden layer are scaled by a scaling layer with scaling parameters $a_1$ and $a_2$ given as

$$N_1(x, \boldsymbol{\omega}, \boldsymbol{b}) = a_1 \tilde{y}(x) \tag{11}$$

$$N_2(x, \boldsymbol{\omega}, \boldsymbol{b}) = a_2 \tilde{y}_1(x) \tag{12}$$

and they can be constants or functions of $x$ to scale up or down $\tilde{y}(x)$ and $\tilde{y}_1(x)$. Introducing these two scale-controllers $a_1$ and $a_2$ is to deal with the cases where the function values of solutions are much less or larger than the order of 1. The outputs of the output layer of the network $N_1$ and $N_2$ are transmitted to the terms $f_{\text{eq1}}$, $f_{\text{eq2}}$, $f_{\text{ic1}}$ and $f_{\text{ic2}}$ in the loss function $f_{\text{loss}}$. The network parameters $\boldsymbol{\omega}$ and $\boldsymbol{b}$ are found by minimizing the $f_{\text{loss}}$.

Similar to [36], substitute formulae (9) and (10) into equation (2) and convert solving the equation (2) into finding the minimum of a optimization problem formulated as

$$\{\boldsymbol{\omega}, \boldsymbol{b}\} = \arg\min_{\{\boldsymbol{\omega}, \boldsymbol{b}\}} f_{\text{loss}} \tag{13}$$

where $f_{\text{loss}}$ is the total loss function given as

$$f_{\text{loss}} = f_{\text{eq1}} + f_{\text{eq2}} + f_{\text{ic1}} + f_{\text{ic2}} \tag{14}$$

where $f_{\text{eq1}}$ is the deviation of the networks $N_1$ and $N_2$ approximating the first sub-equation in equation (8); $f_{\text{eq2}}$ is the deviation of the networks $N_1(x, \boldsymbol{\omega}, \boldsymbol{b})$ and $N_2(x, \boldsymbol{\omega}, \boldsymbol{b})$ approximating the second sub-equation in equation (8); $f_{\text{ic1}}$ is the deviation of the network $N_1(x, \boldsymbol{\omega}, \boldsymbol{b})$ approximating the first initial condition in equation (8); $f_{\text{ic2}}$ is the deviation of the networks $N_2(x, \boldsymbol{\omega}, \boldsymbol{b})$ approximating the second initial condition in equation (8). All these four losses are given as

$$f_{\text{eq1}} = \frac{1}{K}\sum_{i=1}^{K}\left|N_1'(x_i, \boldsymbol{\omega}, \boldsymbol{b}) - N_2(x_i, \boldsymbol{\omega}, \boldsymbol{b})\right|^2 \tag{15}$$

$$f_{\text{eq2}} = \frac{1}{K}\sum_{i=1}^{K}\left|\begin{array}{l} N_2'(x_i, \boldsymbol{\omega}, \boldsymbol{b}) + \dfrac{\alpha}{x} N_2(x_i, \boldsymbol{\omega}, \boldsymbol{b}) \\ +f(x_i, N_1(x_i, \boldsymbol{\omega}, \boldsymbol{b})) - g(x_i) \end{array}\right|^2 \tag{16}$$

$$f_{\text{ic1}} = \left|N_1(0, \boldsymbol{\omega}, \boldsymbol{b}) - c_1\right|^2 \tag{17}$$

$$f_{\text{ic2}} = \left|N_2(0, \boldsymbol{\omega}, \boldsymbol{b}) - c_2\right|^2 \tag{18}$$

where $x_i$ are training data points sampled in the domain $[0, A]$, namely

$$\{x_i\}_{i=1}^{K} \subseteq [0, A] \tag{19}$$

and $K$ is the number of training data points. All the training data points can be user-defined. One can use uniform points or random points or some specific data distributions as one wishes to. To solve the optimization problem (13), Adam solver and L-BFGS-B solver are combined. The Adam solver is first used for a number of epochs and then L-BFGS-B solver is employed.

To further increase the accuracy and mitigate manual intervention in deciding the training data distribution, an adaptive data point sampling strategy is devised. Assume there is another data set which is different from $\{x_i\}_{i=1}^{K}$ and denote it as

$$\{x_j\}_{j=1}^{M} \subseteq [0, A] \tag{20}$$

where $M$ is the number of data points in this set. Over this data set, calculate the values of the following function

$$N_1'(x_i) - N_2(x_i) \tag{21}$$

$$N_2(x_i) + \frac{\alpha}{x} N_2(x_i) + f(x_i, N_1(x_i)) - g(x_i) \tag{22}$$

and copy the data points on which the values of the two functions are larger than a prescribed parameter $\delta$ for example 0.01 from this data set to the training data set. After that, re-train the network over the new training data set.

As mentioned above, there is an issue with using ANN-based methods to solve white-dwarf problem. The root cause for this issue is they are stochastic and this randomness could make $y^2(x)$–$C$ in equation (6) less than 0. As a consequence, $(y^2(x)–C)^{3/2}$ would be complex, which causes errors in the training process since the training algorithm cannot cope with complex values. To deal with this issue, we propose

to use the absolute value of $y^2(x)-C$ and rewrite equation (6) as

$$\begin{cases} y''(x)+\frac{2}{x}y'(x)+\left|y^2(x)-C\right|^{3/2}=0 \\ x\geq 0, \quad y(0)=1, \quad y'(0)=0 \end{cases} \tag{23}$$

for $y(x)$ must be not less than $C$, otherwise the equation (6) will produce complex $y(x)$ which is unphysical. It should be noted that equation (23) is only valid for problem (6) when

$$y(x)\geq\sqrt{C} \tag{24}$$

which can be used to determine the domain length.

### B. ALGORITHM DESCRIPTION

The algorithm of the proposed deep neural network for solving the nonlinear Lane-Emden equation is given below.

*Step 1:* Transform the nonlinear Lane-Emden equation into a system of differential equation of first order;

*Step 2:* Construct the deep neural network given in Fig. 1;

*Step 3:* Sampling the training data points in the interval $[0, A]$ to obtain the training data set $\{x_i\}_{i=1}^K$; sampling the data points in the interval $[0, A]$ to obtain the data set $\{x_j\}_{j=1}^M$;

*Step 4:* Define the total loss function $f_{\text{loss}}$ in formula (14);

*Step 5:* Use Adam solver and L-BFGS-B solver to train the deep neural network over the training data set $\{x_i\}_{i=1}^K$ by solving the optimization problem (13) and obtain the network weights and biases;

*Step 6:* Over the data set $\{x_j\}_{j=1}^M$, compute the formulae (21) and (22) and copy the data points on which the values of the two functions in formulae (21) and (22) are larger than a prescribed parameter $\delta$ from the data set $\{x_j\}_{j=1}^M$ to the training data set $\{x_i\}_{i=1}^K$;

*Step 7:* Over the new training data set, execute the steps 5 and 6 to re-train the network until the maximum of the function values formulae (21) and (22) are less than the parameter $\delta$.

All the training work can be done by using deep learning framework such as Tensorflow and Pytorch in which Adam solver and L-BFGS-B solver are incorporated and the differentiation can be achieved by using automatic differentiation. The formal description of the forgoing procudure is presented in Algorithm 1. The complexity of the proposed model strongly depends on the hyper parameters of the network, such as the depth and width of the hidden layers, and the activation functions used in the network structure, the initialization scheme of the weights and biases and so on. In principle, they are not fixed and need to be adjusted problem by problem. Ignore the benefits from using GPU and advanced deep learning framework such as Tensorflow and Pytorch, the complexity of the proposed model can be approximated by that of fully-connected neural network which has been studied in [46]–[48]. Denote the matrix by $W_{ji}$ which is a matrix with $j$ rows and $i$ columns and contains the weights going from layer $i$ to layer $j$ and one has $K$ training data points. Since there are $L+2$ fully connected layers in the network

**Algorithm 1** Deep-Learning Based Method for Solving Lane-Emden Type Equation With Adaptive Strategy

1. **Preparation**: transform the nonlinear Lane-Emden equation into a system of differential equation of first order.
2. **Input**: The domain $[0, A]$, the differential equation system, the precision parameter $\delta$.
3. **Initialization**: the network parameters $L$, $W$, $\boldsymbol{\omega}$ and $\boldsymbol{b}$; the numbers of the data sets $K$ and $M$ in formulae (19) and (20); $k=0$, $\varepsilon=10\delta$.
4. Sample datasets $\{x_i\}_{i=1}^K$ and $\{x_j\}_{j=1}^M$ in formulae (19) and (20).
5. **While** $\varepsilon>\delta$ **do**
6. use Adam solver and L-BFGS-B solver to train the deep neural network over the training data set $\{x_i\}_{i=1}^K$ by solving the optimization problem (13) for a number of epochs and obtain the network weights $\boldsymbol{\omega}$ and biases $\boldsymbol{b}$.
7. calculate formulae (21) and (22), and $\varepsilon=\max\{\max\{N_1'(x_i)-N_2(x_i)\}_{i=1}^K, \max\{N_2(x_i)+\frac{\alpha}{x}N_1(x_i)+f(x_i,N_1(x_i))-g(x_i)\}_{i=1}^K\}$ and copy the data points on which the values of the two formulae (21) and (22) are larger than $\delta$ from the data set $\{x_j\}_{j=1}^M$ to the training data set $\{x_i\}_{i=1}^K$.
8. $k=k+1$.
9. **end while**
10. **Return**: the network weights $\boldsymbol{\omega}$ and biases $\boldsymbol{b}$.

structure, one needs $L+1$ matrices to represent weights between these layers. Suppose the number of the iterations for the whole training process in the while loop of the Algorithm 1, namely the adaptive training algorithm, is $n_{\text{it}}$ and the while loop executes $n_k$ times, the total time complexity is approximated to be $O\left(n_k n_{\text{it}}\sum_{i=1,j=i+1}^{L+1} j*i*K\right)$ [46]–[48]. The space complexity is dominant by the memory storage of the weight matrices and it can be approximated by $O\left(\sum_{i=1,j=i+1}^{L+1} j*i\right)$. Note that the above-deduced time complexity can be reduced by using GPU and Tensorflow and Pytorch.

## IV. NUMERICAL EXPERIMENTS AND DISCUSSIONS

In this section, numerical experiments will be carried out in the three types of Lane-Emden equations (2), (4) and (23) to validate our proposed method.

### A. EXPERIMENTAL SETUP

The workstation we use is equipped with two Intel Xeon E5-2630v3 CPUs, 128G RAM, one Nvidia Quadro RTX4000 GPU with 8G graphic memory. The integrated development environment we use is Spyder with Python 3.7 and Tensorflow 2.2.

**TABLE 1. Parameters of the experiment setup.**

| Problem | $A$=1 | $A$=10 | $A$=100 | $K$ | $M$ |
|---|---|---|---|---|---|
| C-1) | $m$=0,1,5 | $m$=0,1,5 | $m$=0,1,5 | $10A$ | $100A$ |
| C-2) | / | / | / | $10A$ | $100A$ |
| C-3) | $C$=0,0.1,0.5,0.9 | $C$=0,0.1,0.5,0.9 | $C$=0,0.1,0.5,0.9 | $10A$ | $100A$ |

### B. EXPERIMENTAL CONDITIONS

For small domain with interval length of 1, namely $A = 1$, problems (2), (4) and (6) are solved and the results are compared with numerical results given by Runge-Kutta of fourth order and analytical or approximate analytical solutions if they exist. In problem (2), the polytropic index $m$ is chosen to be 0, 1 and 5. The constant $C$ in problem (6) is chosen to be 0, 0.2, 0.5 and 1. For large domains with interval length of 10 and 100, namely $A = 10$ and 100. Other conditions are the same as that of small domain. The data sets $\{x_i\}_{i=1}^{K}$ and $\{x_j\}_{j=1}^{M}$ are sampled in the domain randomly with $K = 10A$ and $M = 100A$. Parameters of the experiment setup are listed in Table 1.

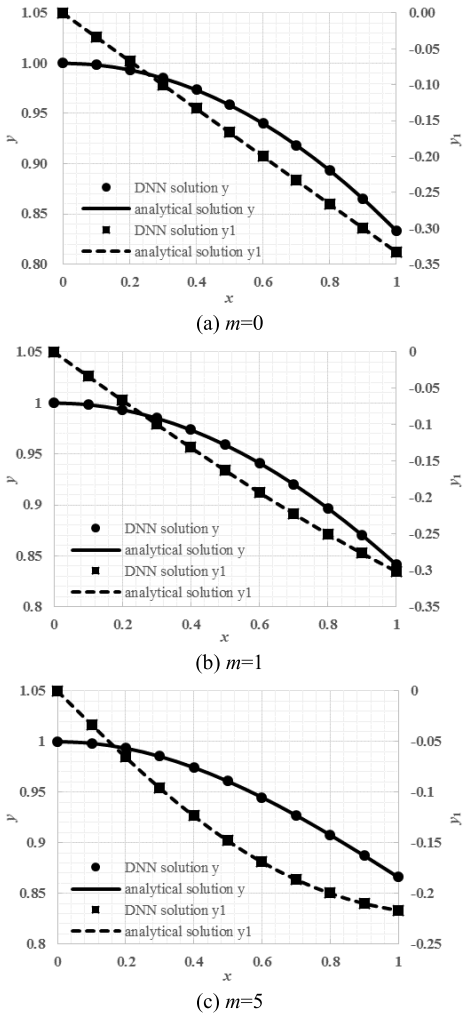
The number of neurons in each layer of the hidden layer is chosen to be $W = 10$ and the number of layers in the hidden layer is chosen to be $L = 5$. There are 2 layers in the output layer. In each layer, the number of neurons is 2. Especially, in the scaling layer of the output layer, the neurons are constants which are different case by case. The activation function is chosen to be tanh. Glorot uniform distribution is used to initialize of the weights and biases in the network. The number of epochs for Adam optimizer in the Tensorflow is chosen to be 10000. It should be noted that the values of these parameters, activation functions and initialization scheme of weights and biases are chosen for the sake of demonstration of the applicability of our method and thus likely not optimal.

### C. RESULTS AND DISCUSSIONS

#### 1) POLYTROPIC GAS SPHERE PROBLEM (THE STANDARD LANE-EMDEN EQUATION)

By use of the proposed deep-learning based method, the solutions to polytropic gas sphere problems with the polytropic index $m = 0$, 1 and 5 and domain length $A = 1$, 10 and 100 are shown in Figure 2, Figure 3 and Figure 4, respectively, where the analytical solutions given by equation (3) are also presented to validate the efficiency of our method. In Figure 4 the solution in the interval [10, 100] is nearly zero and there is no point in calculating it. Hence it is not presented. Also in Figure 4(a), the solutions given by single layer neural network (SNN) is displayed. In this SNN, no adaptive training data sampling strategy and output scaling strategy are used.

Overall, it can be seen from these figures that all the calculated results $y$ produced by our proposed deep neural network (DNN) are in good agreement with the analytical solutions given by equation (3). When $A = 1$, comparisons between our results in Figure 3 and the results in [19], [27] and [28] are made and all maximum absolute errors are listed in Table 2. For $m = 0$, [19], [27] and [28] are able to exactly reproduce the analytical solution given by (3). The maximum



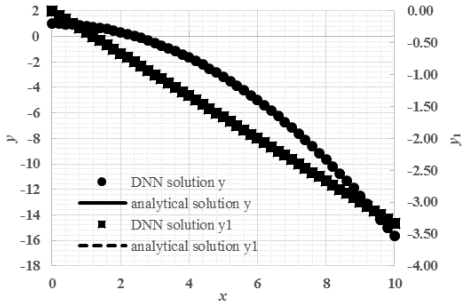

(a) $m$=0

(b) $m$=1

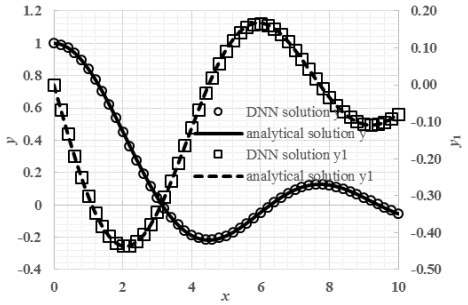
(c) $m$=5

**FIGURE 2. The comparisons between the deep neural network (DNN) solutions and the analytical solutions when the polytropic index $m$=0, 1 and 5 and $A$=1; $y$ being the solution to the equation and $y_1$ being the derivate of $y$.**

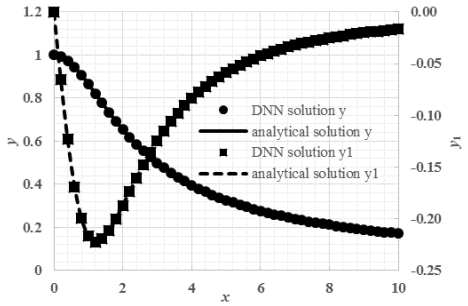
absolute error given by our method is around $1.9 \times 10^{-4}$. For $m = 1$, the maximum absolute error given by [27] and [28] are dependent on the number of basis function used to approximate the solution, and the best error are $2.79 \times 10^{-8}$ with 8 shifted ultraspherical polynomials and $7.48 \times 10^{-11}$ with 8 second kind Chebyshev polynomials, respectively. The maximum absolute error given by our method is around $9.2 \times 10^{-4}$. For $m = 5$, the best maximum absolute error given by [27] is $2.43 \times 10^{-3}$ with 5 shifted ultraspherical polynomials and the maximum absolute error given by [19] and our method are $1.02 \times 10^{-4}$ and $1.9 \times 10^{-4}$, respectively. It can be inferred that they are comparable to those of previous works in some cases. Our method still shows good performance in large domains when $A = 10$ and 100 over which previous works have not demonstrate their methods' abilities. In particular, it can be seen from Figure 4(a) that the solutions produced by SNN deviate from the analytical solutions quite a lot, while those by DNN agree well with the analytical solutions. This demonstrates that our proposed




(a) $m$=0




(b) $m$=1




(c) $m$=5

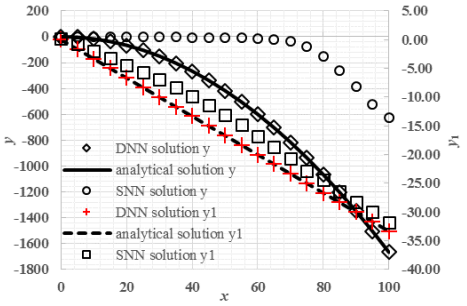
**FIGURE 3.** The comparisons between the deep neural network (DNN) solutions and the analytical solutions when the polytropic index $m$=0, 1 and 5 and $A$=10; $y$ being the solution to the equation and $y_1$ being the derivate of $y$.

ANN works better in cases where the function values are much larger than the order of 1.

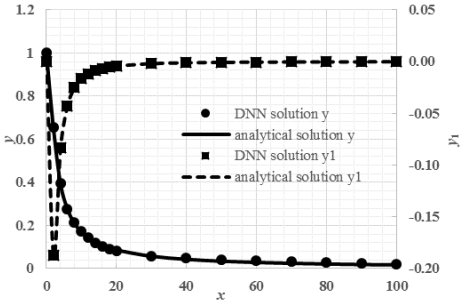
In addition to the solutions $y$ to the Lane-Emden equations, their derivative functions $y_1$ are obtained accompanied with $y$. They are presented in Figure 2, Figure 3 and Figure 4 and compared with the derivatives of the analytical solutions. It can also be observed from these figures that all the calculated results $y_1$ produced by our proposed deep neural network (DNN) are in good agreement with the derivatives of the analytical solutions given by equation (3). This agreement can also confirm the reliability of the solutions $y$, since it reflects that the local variations of $y$ are also agrees well with those of analytical solutions. The previous works do not calculate the derivative functions. We are not able to compare our results with those.

### 2) ISOTHERMAL GAS SPHERE PROBLEM

By use of the proposed deep-learning based method, the solutions to isothermal gas sphere problems with domain




(a) $m$=0




(b) $m$=5

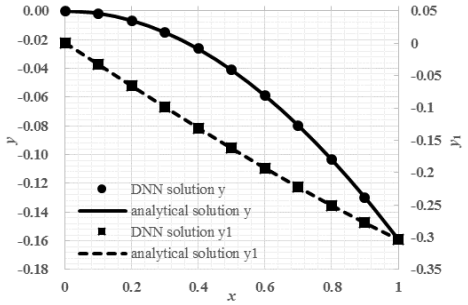
**FIGURE 4.** The comparisons between the deep neural network (DNN) solutions and the analytical solutions when the polytropic index $m$=0 and 5 and $A$=100, in (a) the solutions given by single lay neural network (SNN) being displayed; $y$ being the solution to the equation and $y_1$ being the derivate of $y$.

**TABLE 2.** Comparison of maximum absolute errors of previous works and our method for $A$=1.

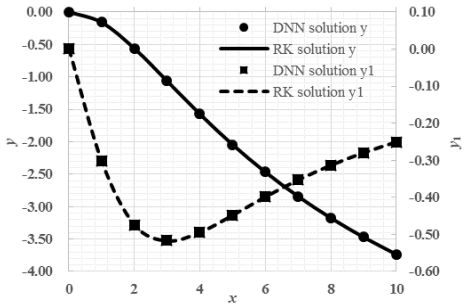
| $m$ | [19] | [27] | [28] | This work |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1.9e-4 |
| 1 | / | 2.79e-8 | 7.48e-11 | 9.2e-4 |
| 5 | 1.02e-4 | 2.43e-3 | / | 1.9e-4 |

length $A = 1$, 10 and 100 are shown in Fig. 5 where the analytical solutions given by equation (5) for $A = 1$ and the numerical solutions given by fourth order Runge-Kutta method for $A = 10$ and 100 are also presented to validate the efficiency of our method. On the whole, it can be seen from Figure 5 that the solutions $y_1$ given by our DNN based method are inconsistent with the analytical solutions and numerical solutions given by Runge-Kutta method, even in large domains. This again demonstrates our method are capable of giving correct solutions in large domain. The maximum absolute errors of $y$ by our method for $A = 1$, 10 and 100 are listed in Table 3. It can be seen that all the maximum absolute errors of $y$ are less than 0.005. A further comparison between our results in Figure 5 and the results in [19], [27] and [28] is made when $A = 1$ and all maximum absolute errors are listed in Table 4. The maximum absolute errors of previous works are $3.2 \times 10^{-7}$, $6.8 \times 10^{-5}$ and $2.5 \times 10^{-10}$, respectively, which are smaller than $2.9 \times 10^{-4}$ in our work. But the maximum absolute errors of our results are acceptable. Since no previous works report the results when $A = 10$ and 100, we only presents the maximum absolute errors in this work and they are $5.6 \times 10^{-3}$ and $5.8 \times 10^{-3}$, respectively.

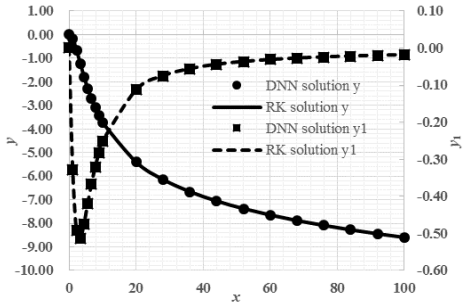
In addition to the solutions $y$ to the Lane-Emden equations, their derivative functions $y_1$ are obtained accompanied




(a) $A$=1




(b) $A$=10




(c) $A$=100

**FIGURE 5.** The comparisons between the deep neural network (DNN) solutions and the analytical solutions when $A$=1 or the numerical solutions given by Runge-Kutta (RK) method when $A$=10 and 100; $y$ being the solution to the equation and $y_1$ being the derivate of $y$.

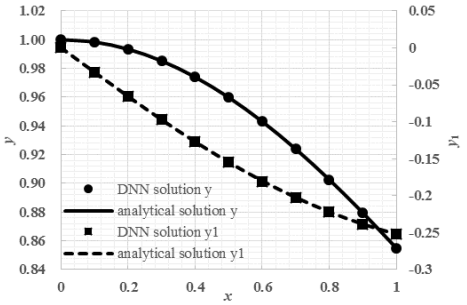
**TABLE 3.** Maximum absolute errors of the solutions $y$ given by our method for $A$=1, 10 and 100.

| $A$ | 1 | 10 | 100 |
|---|---|---|---|
| error | 2.9e-4 | 8.9e-4 | 3.9e-3 |

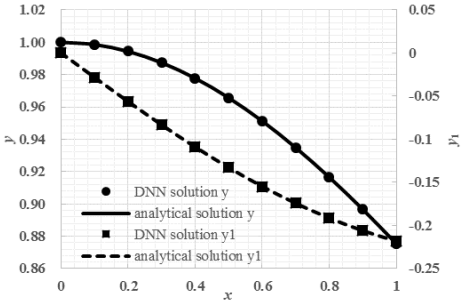
with $y$. They are presented in Figure 5 and compared with the derivatives of the numerical solutions given by Runge-Kutta method. It can also be observed from this figure that all the calculated results $y_1$ produced by our method are in good agreement with the derivatives of the numerical solutions. This agreement can again confirm the reliability of the solutions $y$ for the same reason stated above. The previous works do not calculate the derivative functions. We are not able to compare our results with those.

### 3) WHITE-DWARF STAR PROBLEM

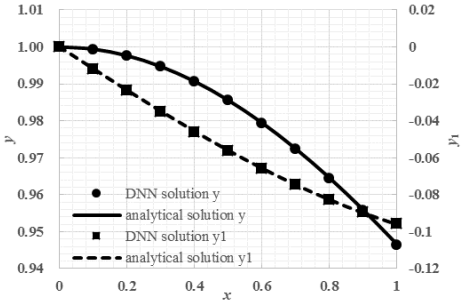
By use of the proposed deep-learning based method, the solutions to isothermal gas sphere problems with different values of $C = 0, 0.1, 0.5$ and 0.9 and different domain length



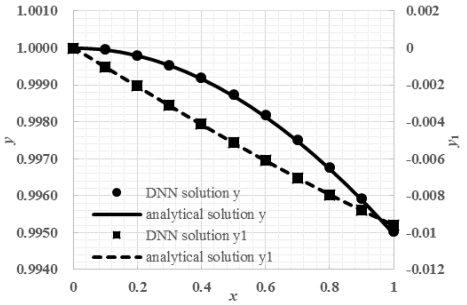

(a) $C$=0




(b) $C$=0.1



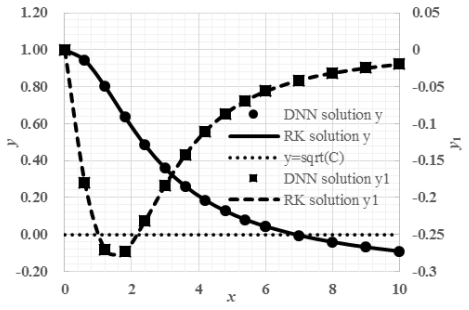

(c) $C$=0.5



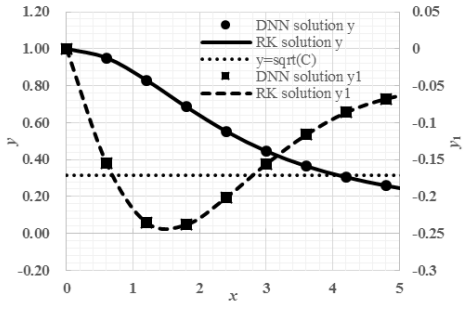

(d) $C$=0.9

**FIGURE 6.** The comparisons between the deep neural network (DNN) solutions and the analytical solutions when $A$=1; $y$ being the solution to the equation and $y_1$ being the derivate of $y$.

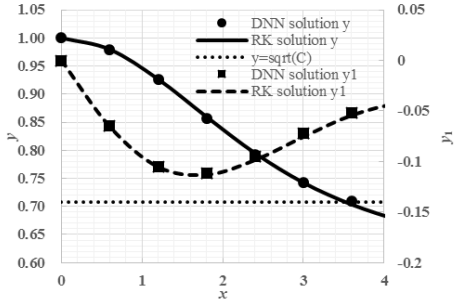
$A = 1$ and 10 are shown in Figure 6 and Figure 7, respectively, where the analytical solutions given by equation (23) for $A = 1$ and the numerical solutions given by fourth order Runge-Kutta method for $A = 10$ are also presented to validate the efficiency of our method. It should be noted that the maximum valid $A$ is achieved for equation (23) and is calculated to be around 7 by using equation (24) when $C = 0$. Hence the domain length is set to be 10 and the critical $y$ where $y = \sqrt{C}$ is plotted in dotted line in Figure 7. In other word, the part



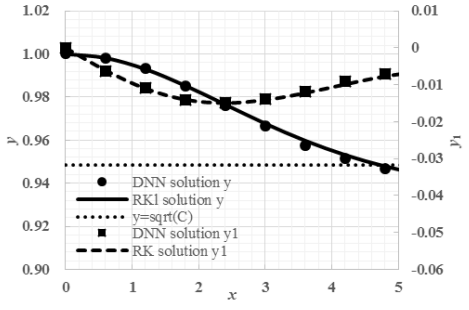

(a) $C$=0




(b) $C$=0.1




(c) $C$=0.5




(d) $C$=0.9

**FIGURE 7.** The comparisons between the deep neural network (DNN) solutions and the numerical solutions given by Runge-Kutta method when $A$=10; $y$ being the solution to the equation and $y_1$ being the derivate of $y$; the dotted line representing the line $y = \sqrt{C}$ above which $y$ ($x$) is physically reasonable.

of curve being above this dotted line is valid and physically reasonable. Only the physically reasonable part of the curves are plotted in the Figure 7. It can be seen from Figure 6 and Figure 7 that the solutions $y$ given by our DNN based method are inconsistent with the analytical solutions and numerical solutions given by Runge-Kutta method, whether in physically reasonable domains or unphysical ones. The maximum

**TABLE 4.** Comparison of maximum absolute errors of the solutions $y$ given by previous works and our method for $A$=1.

| Reference | [19] | [27] | [28] | This work |
|---|---|---|---|---|
| error | 3.2e-7 | 6.8e-5 | 2.5e-10 | 2.9e-4 |

**TABLE 5.** Maximum absolute errors of the solution $y$ given by our method for $A$=1 and 10.

| $A$ | $C$=0 | $C$=0.1 | $C$=0.5 | $C$=0.9 |
|---|---|---|---|---|
| 1 | 3.4e-3 | 7.1e-5 | 2.4e-4 | 6.9e-5 |
| 10 | 3.6e-4 | 8.1e-4 | 6.5e-3 | 2.0e-3 |

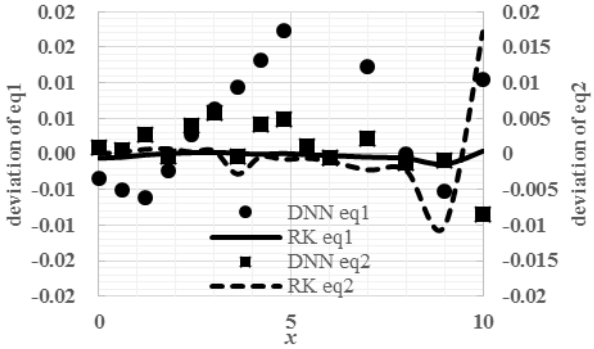



**FIGURE 8.** The comparisons between the deep neural network (DNN) and Runge-Kutta method when $A$=10. The $eq_1$ and $eq_2$ denote the results of formulae (21) and (22) calculated by using $y$ and $y_1$ from Figure 7(c).

absolute errors of $y$ by our method for $A = 1$ and 10 are listed in Table 5. It can be seen that all the maximum absolute errors of $y$ are less than 0.007. In addition to the solutions $y$ to the Lane-Emden equations, their derivative functions $y_1$ are obtained accompanied with $y$. They are presented in Figure 6 and Figure 7 and compared with the derivatives of the analytical solutions. It can also be observed from these figures that all the calculated results $y_1$ produced by the deep neural network (DNN) are in good agreement with the derivatives of the analytical solutions. This agreement can again confirm the reliability of the solutions $y$ for the same reason stated above.

### 4) DISCUSSIONS

In this subsection, the benefits and tradeoffs of the proposed solution approach over the baseline (Runge-Kutta (RK)). The proposed solution approach does have some benefits over RK. For example, it provides a closed-form representation of the solution that is infinitely differentiable, and unlike RK tabular solutions, the solution given by our method is valid over the entire domain, eliminating the need for interpolation. Our method are able to be extended to solve 2-D, 3-D and 4-D Lane-Emden type equations and lots of works have shown that deep neural network has superiority in solving high dimensional problems, while RK can only deal with 1-D problems. Here the 2-D, 3-D and 4-D problems can be time and space dependent. It should be admitted that RK has higher precision than our method in many cases. This is partially due to using single precision in the computation to utilize the superiority of GPU's single precision computability over its double precision computability and single

precision computation is commonly used in field of deep learning. But we can show that in some cases our method are comparable or even better than RK. The results presented in the Figure 8 is an example. The $eq_1$ and $eq_2$ denote the results of formulae (21) and (22) calculated by using $y$ and $y_1$ from Figure 7(c). They are in fact the deviations of the right hand side of the first two sub-equations in equation (8) from 0. As can be seen from Figure 8 that, although the deviation of $eq_1$ for DNN is larger than that for RK, the deviation of $eq_2$ for DNN is clearly less than that for RK when $x$ is close to 10. This demonstrates that DNN solution can be better than RK solution in some subset of the domain.

## V. CONCLUSION

In this article, a deep-learning based calculation method is proposed to solve the nonlinear Lane-Emden type equation. The output layer of this network has two layers, the last one of which scaling the outputs in order to deal with the cases where the values of the functions approximated by the network is much less or larger than the order of 1. The Lane-Emden equation and its initial conditions are employed to construct the loss function, and the problem of solving Lane-Emden equation is transformed into an optimization problem. There is no need for one to carefully design the trial solution case by case. An adaptive data sampling strategy is proposed to increase the accuracy. A reformulation of the white-dwarf equation is proposed to allow ANN-based methods to be employed to solve it. This reformulation is useful for any stochastic methods designing for solving this equation. The activation functions in our network and the training data points are user-defined and the points are not necessarily equidistant in the domain. Numerical experiments demonstrate that the proposed method is effective in both small domains and large ones where the existent ANN-based methods sometimes fail to given promising results. More future work can be done to improve our method, such as conducting a rigorous mathematical proof of the proposed method, applying the deep domain decomposition method proposed in [45] to deal with large domain problems and extending the method to high dimensional Lane-Emden type equations.

## NOTATION TABLE

| | |
|---|---|
| $A$ | the length of the domain |
| $a_1$ | the scale-controller in the equation (11) |
| $a_2$ | the scale-controller in the equation (12) |
| $\boldsymbol{b}$ | the bias vector of deep neural network |
| $C$ | the constant in the white-dwarf star problem |
| $c_1$ | the constant in the equation (1) |
| $c_2$ | the constant in the equation (1) |
| $f(x, y)$ | the function in the equation (1) |
| $f_{\text{loss}}$ | the loss function of deep neural network |
| $f_{\text{eq1}}$ | the deviation of the network $N_1(x, \boldsymbol{\omega}, \boldsymbol{b})$ approximating the first sub-equation in equation (8) |
| $f_{\text{eq2}}$ | the deviation of the networks $N_2(x, \boldsymbol{\omega}, \boldsymbol{b})$ approximating the second sub-equation in equation (8) |
| $f_{\text{ic1}}$ | the deviation of the network $N_1(x, \boldsymbol{\omega}, \boldsymbol{b})$ approximating the first initial condition in equation (8) |
| $f_{\text{ic2}}$ | the deviation of the networks $N_2(x, \boldsymbol{\omega}, \boldsymbol{b})$ approximating the second initial condition in equation (8) |
| $g(x)$ | the function in the equation (1) |
| $K$ | the number of training data points |
| $L$ | the depth of the hidden layer |
| $M$ | the number of data points in the dataset given by formula (20) |
| $m$ | the polytroic index in the equation (2) |
| $\boldsymbol{N}(x, \boldsymbol{\omega}, \boldsymbol{b})$ | Deep neural network with weight vector $\boldsymbol{\omega}$ and bias vector $\boldsymbol{b}$ and two output terminals |
| $N_1(x, \boldsymbol{\omega}, \boldsymbol{b})$ | the first output of $\boldsymbol{N}(x, \boldsymbol{\omega}, \boldsymbol{b})$ |
| $N_1'(x, \boldsymbol{\omega}, \boldsymbol{b})$ | the partial derivative of $\boldsymbol{N}_1(x, \boldsymbol{\omega}, \boldsymbol{b})$ with respect to $x$ |
| $N_2(x, \boldsymbol{\omega}, \boldsymbol{b})$ | the second output of $\boldsymbol{N}(x, \boldsymbol{\omega}, \boldsymbol{b})$ |
| $N_2'(x, \boldsymbol{\omega}, \boldsymbol{b})$ | the partial derivative of $\boldsymbol{N}_2(x, \boldsymbol{\omega}, \boldsymbol{b})$ with respect to $x$ |
| $O(\bullet)$ | the order of complexity |
| $q$ | a constant given by $(1 - C)^{1/2}$ |
| $s$ | the activation function in the deep neural network |
| $W$ | the width of the hidden layer |
| $x$ | the independent variable the Lane-Emden type equation and the element in the domain |
| $\{x_i\}_{i=1}^{K}$ | the training data set |
| $\{x_j\}_{j=1}^{M}$ | the dataset given by formula (20) |
| $y$ | the solution to the Lane-Emden type equation and the element in the range |
| $\tilde{y}(x)$ | the first output of the hidden layer |
| $\tilde{y}_1(x)$ | the second output of the hidden layer |
| $y'$ | the first order derivative of $y$ with respect to $x$ |
| $y_1$ | the first order derivative of $y$ with respect to $x$; the same as $y'$ |
| $y''$ | the second order derivative of $y$ with respect to $x$ |
| $\alpha$ | the constant in the equation (1) |
| $\delta$ | the precision parameter for the adaptive sampling |
| $\boldsymbol{\omega}$ | the weight vector of deep neural network |

## REFERENCES

[1] J. H. Lane, "On theoretical temperature of the sun under the hypothesis of a gaseous mass maintaining its internal heat and depending on the laws of gases known to terrestrial experiment," *Amer. J. Sci. Arts.*, vol. 148, pp. 57–74, Jul. 1870.

[2] S. Aydinlik and A. Kiris, "A high-order numerical method for solving nonlinear lane-emden type equations arising in astrophysics," *Astrophys. Space Sci.*, vol. 363, no. 12, Dec. 2018, Art. no. 264.

[3] S. Chandrasekhar, *Introduction to Study of Stellar Structure*. Chicago, IL, USA: Univ. of Chicago Press, 1967, pp. 220–266.
[4] L. D. Zhu and J. H. Wang, "The formation and propagation of shock formed during gravitational collapse of a gaseous polytrope," *Acta Astrophy. Sinica*, vol. 14, no. 4, pp. 358–368, 1994.
[5] P. L. Chambré, "On the solution of the Poisson-Boltzmann equation with application to the theory of thermal explosions," *J. Chem. Phys.*, vol. 20, no. 11, pp. 1795–1797, Nov. 1952.
[6] R. C. Duggan and A. M. Goodman, "Pointwise bounds for a nonlinear heat conduction model of the human head," *Bull. Math. Biol.*, vol. 48, no. 2, pp. 229–236, Mar. 1986.
[7] R. W. Dickey, "Rotationally symmetric solutions for shallow membrane caps," *Quart. Appl. Math.*, vol. 47, no. 3, pp. 58–571, 1989.
[8] N. T. Shawagfeh, "Nonperturbative approximate solution for Lane–Emden equation," *J. Math. Phys.*, vol. 34, no. 9, pp. 4364–4369, Sep. 1993.
[9] A. Wazwaz, "A new algorithm for solving differential equations of Lane-Emden type," *Appl. Math. Comput.*, vol. 118, pp. 287–310, Mar. 2001.
[10] S. Liao, "On the homotopy analysis method for nonlinear problems," *Appl. Math. Comput.*, vol. 147, no. 2, pp. 499–513, Jan. 2004.
[11] O. P. Singh, R. K. Pandey, and V. K. Singh, "An analytic algorithm of Lane–Emden type equations arising in astrophysics using modified homotopy analysis method," *Comput. Phys. Commun.*, vol. 180, no. 7, pp. 1116–1124, Jul. 2009.
[12] S. Iqbal and A. Javed, "Application of optimal homotopy asymptotic method for the analytic solution of singular Lane–Emden type equation," *Appl. Math. Comput.*, vol. 217, no. 19, pp. 7753–7761, Jun. 2011.
[13] J. H. He, "Homotopy perturbation technique," *Comput. Methods Appl. Mech. Eng.*, vol. 178, nos. 3–4, pp. 257–262, 1999.
[14] M. S. H. Chowdhury and I. Hashim, "Solutions of a class of singular second-order IVPs by homotopy-perturbation method," *Phys. Lett. A*, vol. 365, nos. 5–6, pp. 439–447, Jun. 2007.
[15] J. I. Ramos, "Series approach to the Lane-Emden equation and comparison with the homotopy perturbation method," *Chaos, Solitons Fractals*, vol. 38, no. 2, pp. 400–408, 2008.
[16] R. Torres-Córdoba and E. A. Martínez-García, "Exact analytic solution of an unsolvable class of first Lane–Emden equation for polytropic gas sphere," *New Astron.*, vol. 82, Jan. 2021, Art. no. 101458.
[17] C. Tsitouras, "Explicit Runge–Kutta methods for starting integration of Lane–Emden problem," *Appl. Math. Comput.*, vol. 354, pp. 353–364, Aug. 2019.
[18] S. A. Yousefi, "Legendre wavelets method for solving differential equations of Lane–Emden type," *Appl. Math. Comput.*, vol. 181, no. 2, pp. 1417–1422, Oct. 2006.
[19] Y. H. Youssri, W. M. Abd-Elhameed, and E. H. Doha, "Ultraspherical wavelets method for solving Lane–Emden type equations," *Rom. J. Phys.*, vol. 60, nos. 9–10, pp. 1298–1314, 2015.
[20] R. Singh, J. Shahni, H. Garg, and A. Garg, "Haar wavelet collocation approach for lane-emden equations arising in mathematical physics and astrophysics," *Eur. Phys. J. Plus*, vol. 134, no. 11, Nov. 2019, Art. no. 548.
[21] K. Parand, M. Dehghan, A. R. Rezaei, and S. M. Ghaderi, "An approximation algorithm for the solution of the nonlinear Lane–Emden type equations arising in astrophysics using Hermite functions collocation method," *Comput. Phys. Commun.*, vol. 181, no. 6, pp. 1096–1108, Jun. 2010.
[22] M. Dehghan, S. Aryanmehr, and M. R. Eslahchi, "A technique for the numerical solution of initial-value problems based on a class of Birkhoff-type interpolation method," *J. Comput. Appl. Math.*, vol. 244, pp. 125–139, May 2013.
[23] K. Parand, A. R. Rezaei, and A. Taghavi, "Lagrangian method for solving Lane–Emden type equation arising in astrophysics on semi-infinite domains," *Acta Astronautica*, vol. 67, nos. 7–8, pp. 673–680, Oct. 2010.
[24] H. Singh, "An efficient computational method for the approximate solution of nonlinear lane-emden type equations arising in astrophysics," *Astrophys. Space Sci.*, vol. 363, no. 4, Apr. 2018, Art. no. 71.
[25] R. K. Pandey and N. Kumar, "Solution of Lane–Emden type equations using bernstein operational matrix of differentiation," *New Astron.*, vol. 17, no. 3, pp. 303–308, Apr. 2012.
[26] R. K. Pandey, N. Kumar, A. Bhardwaj, and G. Dutta, "Solution of Lane–Emden type equations using Legendre operational matrix of differentiation," *Appl. Math. Comput.*, vol. 218, no. 14, pp. 7629–7637, Mar. 2012.
[27] E. H. Doha, W. M. Abd-Elhameed, and Y. H. Youssri, "Second kind Chebyshev operational matrix algorithm for solving differential equations of Lane–Emden type," *New Astron.*, vols. 23–24, pp. 113–117, Oct. 2013.
[28] W. M. Abd-Elhameed, Y. Youssri, and E. H. Doha, "New solutions for singular Lane–Emden equations arising in astrophysics based on shifted ultraspherical operational matrices of derivatives," *Compu. Meth. Different. Equa.*, vol. 2, no. 3, pp. 171–185, 2014.
[29] B. Căruntu and C. Bota, "Approximate polynomial solutions of the nonlinear Lane–Emden type equations arising in astrophysics using the squared remainder minimization method," *Comput. Phys. Commun.*, vol. 184, no. 7, pp. 1643–1648, Jul. 2013.
[30] A. Ghorbani and M. Bakherad, "A variational iteration method for solving nonlinear Lane–Emden problems," *New Astron.*, vol. 54, pp. 1–6, Jul. 2017.
[31] P. Roul, "A new mixed MADM-collocation approach for solving a class of Lane–Emden singular boundary value problems," *J. Math. Chem.*, vol. 57, no. 3, pp. 945–969, Mar. 2019.
[32] M. Singh and A. K. Verma, "An effective computational technique for a class of Lane–Emden equations," *J. Math. Chem.*, vol. 54, no. 1, pp. 231–251, Jan. 2016.
[33] M. Pathak and P. Joshi, "Application of a coupled approach for the solution of nonlinear singular initial value problems of Lane–Emden type," *Astrophys. Space Sci.*, vol. 363, no. 9, Sep. 2018, Art. no. 191.
[34] A. H. Hadian-Rasanan, D. Rahmati, S. Gorgin, and K. Parand, "A single layer fractional orthogonal neural network for solving various types of Lane–Emden equation," *New Astron.*, vol. 75, Feb. 2020, Art. no. 101307.
[35] A. Verma and M. Kumar, "Numerical solution of Lane–Emden type equations using multilayer perceptron neural network method," *Int. J. Appl. Comput. Math.*, vol. 5, no. 5, Oct. 2019, Art. no. 141.
[36] Z. Sabir, H. A. Wahab, M. Umar, M. G. Sakar, and M. A. Z. Raja, "Novel design of Morlet wavelet neural network for solving second order Lane–Emden equation," *Math. Comput. Simul.*, vol. 172, pp. 1–14, Jun. 2020.
[37] K. M. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. CVPR*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[38] M. Pak and S. Kim, "A review of deep learning in image recognition," in *Proc. 4th Int. Conf. Comput. Appl. Inf. Process. Technol. (CAIPT)*, Kuta Bali, Indonesia, Aug. 2017, pp. 1–3.
[39] L. Khelifi and M. Mignotte, "Deep learning for change detection in remote sensing images: Comprehensive review and meta-analysis," *IEEE Access*, vol. 8, pp. 126385–126400, 2020.
[40] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing [Review Article]," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.
[41] G. Qian, Z. Li, C. He, X. Li, and X. Ding, "Power allocation schemes based on deep learning for distributed antenna systems," *IEEE Access*, vol. 8, pp. 31245–31253, 2020.
[42] Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward Tomorrow's intelligent network traffic control systems," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2432–2455, May 2017.
[43] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, Feb. 2019.
[44] J. Sirignano and K. Spiliopoulos, "DGM: A deep learning algorithm for solving partial differential equations," *J. Comput. Phys.*, vol. 375, pp. 1339–1364, Dec. 2018.
[45] K. Li, K. Tang, T. Wu, and Q. Liao, "D3M: A deep domain decomposition method for partial differential equations," *IEEE Access*, vol. 8, pp. 5283–5294, 2020.
[46] K. Fredenslund. *Computational Complexity of Neural Networks*. Accessed: Oct. 22, 2020. [Online]. Available: https://kasperfred.com/series/introduction-to-neural-networks/computational-complexity-of-neural-networks
[47] M. K. Akhgary. *What is the Time Complexity for Training a Neural Network Using Back-Propagation?* Accessed: Oct. 22, 2020. [Online]. Available: https://ai.stackexchange.com/questions/5728/what-is-the-time-complexity-for-training-a-neural-network-using-back-propagation/20281#20281
[48] A. Lockett. *How Do I Determine the Computational Time Complexity (Big-O) of Back Propagation in a Feedforward Neural Network?* Accessed: Oct. 22, 2020. [Online]. Available: https://www.quora.com/How-do-I-determine-the-computational-time-complexity-Big-O-of-back-propagation-in-a-feedforward-neural-network